

```

EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFFFFFFFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFFFFFFFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDDDD FFFFFFFFFFFFFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEEEEEEEEEEEEEEEEE DDD DDD FFFFFFFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDD DDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD FFF
EEEEEEEEEEEEEEEEEE DDDDDDDDDDDDD FFF

```

Va  
--  
00  
00  
00  
00  
00  
00  
00  
00  
7F  
7F  
7F  
7F  
7F  
7F  
7F  
7F  
7F

EEEEEEEEEE	DDDDDDDD	FFFFFFFFFF	VV	VV	AAAAAA	LL	UU	UU	EEEEEEEEEE
EEEEEEEEEE	DDDDDDDD	FFFFFFFFFF	VV	VV	AAAAAA	LL	UU	UU	EEEEEEEEEE
EE	DD	DD	VV	VV	AA	LL	UU	UU	EE
EE	DD	DD	VV	VV	AA	LL	UU	UU	EE
EE	DD	DD	VV	VV	AA	LL	UU	UU	EE
EEEEEEEEEE	DD	DD	VV	VV	AA	LL	UU	UU	EEEEEEEEEE
EEEEEEEEEE	DD	DD	VV	VV	AA	LL	UU	UU	EEEEEEEEEE
EE	DD	DD	VV	VV	AAAAAAA	LL	UU	UU	EE
EE	DD	DD	VV	VV	AAAAAAA	LL	UU	UU	EE
EE	DD	DD	VV	VV	AA	LL	UU	UU	EE
EE	DD	DD	VV	VV	AA	LL	UU	UU	EE
EEEEEEEEEE	DDDDDDDD	FF	VV	VV	AA	LLLLLLLLLL	UUUUUUUUUU	UUUUUUUUUU	EEEEEEEEEE
EEEEEEEEEE	DDDDDDDD	FF	VV	VV	AA	LLLLLLLLLL	UUUUUUUUUU	UUUUUUUUUU	EEEEEEEEEE

PPPPPPPP	AAAAAA	SSSSSSSS
PPPPPPPP	AAAAAA	SSSSSSSS
PP	AA	SS
PP	AA	SS
PP	AA	SS
PP	AA	SS
PPPPPPPP	AA	SSSSSS
PPPPPPPP	AA	SSSSSS
PP	AAAAAAA	SS
PP	AAAAAAA	SS
PP	AA	SS
PP	AA	SS
PP	AA	SSSSSSSS
PP	AA	SSSSSSSS

{ ++

FILE: SRC\$:EDFVALUE.PAS - Pascal include file to define  
initial values of selected top-level variables.

```
*****
*
*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*  ALL RIGHTS RESERVED.
*
*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
*  TRANSFERRED.
*
*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
*  CORPORATION.
*
*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*
*****
```

FACILITY: VAX/VMS EDF (EDIT/FDL) UTILITY

ABSTRACT: This facility is used to create, modify, and optimize  
FDL specification files.

ENVIRONMENT: NATIVE/USER MODE

AUTHOR: Ken F. Henderson Jr.

CREATION DATE: 27-Mar-1981

MODIFIED BY:

V03-009	KFH0009	Ken Henderson	10 Sep 1983
		Support named UICs.	
V03-008	KFH0008	Ken Henderson	9 Aug 1983
		Fix max value of CLUSTER_SIZE.	
		Fix default of QTAB[TEST_PRIMARY].	
V03-007	KFH0007	Ken Henderson	30 Jul 1983
		Fix SEC TYPE table for audit_trail.	
		Add DEFERRED_WRITE.	

V03-006	KFH0006	Ken Henderson	26 Apr 1983
	Fix various defaults in QTAB. Transferred some initializations to the EDFVAR declarations.		
V03-005	KFH0005	Ken Henderson	14 Apr 1983
	Changed max bucket size to 63 from 65. Added ANALYSIS, OUTPUT, RESPONSES, PROMPTING, SET FUNCTION, GRANULARITY. Added support for SEGMENTED keys.		
V03-004	KFH0004	Ken Henderson	7 Mar 1983
	Changed max bucket size to 65 from 127.		
V03-003	KFH0003	Ken Henderson	11 Sept 1982
	Added initialization of VDATA and BDATA.		
V03-002	KFH0002	Ken Henderson	9 Sept 1982
	Added initialization of QTAB.		
V03-001	KFH0001	Ken Henderson	23-Mar-1982
	Took out reference to EDITFDL_STRING		

-- }



[illegible]

```
SDATA      := (
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL),
    (O,DSC$K_DTYPE_T,DSC$K_CLASS_D,NIL)
);
```

```

VDATA
:= (
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE,
FALSE)

```



);

```
{ +
Initialize the sequencing array.
- }
```

```
PRI_SEQ := (
    15,      { DUMMY_PRIMARY$ }
    8,       { ACCESS, }
    4,       { ACL, }
    13,      { ANALYSIS_OF_AREA, }
    14,      { ANALYSIS_OF_KEY, }
    11,      { AREA, }
    10,      { CONNECT, }
    4,       { DATE, }
    3,       { FILE$, }
    1,       { IDENT, }
    6,       { JOURNAL, }
    12,      { KEY, }
    7,       { RECORD$, }
    9,       { SHARING, }
    2,       { SYSTEM, }
    0,       { TITLE }
);
```

```
{ +
Initialize the 'width' arrays - that indicate how long a particular
keyword should be printed.
- }
```

```
PRIMARY_WIDTH := (
    0,      { DUMMY_PRIMARY$ }
    6,      { ACCESS, }
    3,      { ACL, }
    16,     { ANALYSIS_OF_AREA, }
    15,     { ANALYSIS_OF_KEY, }
    4,      { AREA, }
    7,      { CONNECT, }
    4,      { DATE, }
    4,      { FILE$, }
    5,      { IDENT, }
    7,      { JOURNAL, }
    3,      { KEY, }
    6,      { RECORD$, }
    7,      { SHARING, }
    6,      { SYSTEM, }
    5,      { TITLE }
);
```

```
SECONDARY_WIDTH := (
```

```
{ RESERVE 0 }      0,      { DUMMY_SECONDARY$, }
{ ACCESS PRIMARY }
                    8,      { BLOCK_IO$ }
```

6. { DELETES }  
 3. { GETS }  
 3. { PUTS }  
 9. { RECORD\_IO\$ }  
 8. { TRUNCATES }  
 6. { UPDATES }

## { ACL PRIMARY }

5. { ENTRY }

## { ANALYSIS\_OF\_AREA PRIMARY }

15. { RECLAIMED\_SPACE }

## { ANALYSIS\_OF\_KEY PRIMARY }

9. { DATA\_FILLS, }  
 20. { DATA\_KEY\_COMPRESSION, }  
 23. { DATA\_RECORD\_COMPRESSION, }  
 17. { DATA\_RECORD\_COUNT, }  
 19. { DATA\_SPACE\_OCCUPIED, }  
 9. { DELETIONS, }  
 5. { DEPTH, }  
 19. { DUPLICATES\_PER\_SIDR, }  
 17. { INDEX\_COMPRESSION, }  
 10. { INDEX\_FILLS, }  
 20. { INDEX\_SPACE\_OCCUPIED, }  
 19. { LEVELT\_RECORD\_COUNT }  
 16. { MEAN\_DATA\_LENGTH, }  
 17. { MEAN\_INDEX\_LENGTH, }  
 15. { RANDOM\_ACCESSES, }  
 14. { RANDOM\_INSERTS, }  
 19. { SEQUENTIAL\_ACCESSES, }

## { AREA PRIMARY }

10. { ALLOCATIONS, }  
 19. { BEST\_TRY\_CONTIGUOUS, }  
 11. { BUCKET\_SIZES, }  
 10. { CONTIGUOUS, }  
 17. { EXACT\_POSITIONINGS, }  
 9. { EXTENSIONS, }  
 8. { POSITIONS, }  
 6. { VOLUMES, }

## { CONNECT PRIMARY }

12. { ASYNCHRONOUS }  
 8. { BLOCK\_IO }  
 11. { BUCKET\_CODE }  
 7. { CONTEXT }  
 11. { END\_OF\_FILE }  
 12. { FILE\_BUCKETS }  
 11. { FAST\_DELETE }  
 16. { KEY\_OF\_REFERENCE }  
 17. { KEY\_GREATER\_EQUAL }



```

16. { KEY_GREATER_THAN }
9.  { KEY_LIMIT }
11. { LOCATE_MODE }
12. { LOCK_ON_READ }
13. { LOCK_ON_WRITE }
16. { MANUAL_UNLOCKING }
16. { MULTIBLOCK_COUNT }
17. { MULTIBUFFER_COUNT }
6.  { NOLOCK }
18. { NONEXISTENT_RECORD }
10. { READ_AHEAD }
15. { READ_REGARDLESS }
14. { TIMEOUT_ENABLE }
14. { TIMEOUT_PERIOD }
15. { TRUNCATE_ON_PUT }
19. { TT_CANCEL_CONTROL_0 }
15. { TT_UPCASE_INPUT }
9.  { TT_PROMPT }
16. { TT_PURGE_TYPE_AHEAD }
14. { TT_READ_NOECHO }
16. { TT_READ_NOFILTER }
9.  { UPDATE_IF }
15. { WAIT_FOR_RECORD }
12. { WRITE_BEHIND }

```

## ( DATE PRIMARY )

```

6.  { BACKUPS, }
8.  { CREATIONS, }
10. { EXPIRATIONS, }
8.  { REVISIONS, }

```

## ( FILE PRIMARY )

```

10. { ALLOCATION, }
19. { BEST_TRY_CONTIGUOUS, }
11. { BUCKET_SIZE, }
12. { CLUSTER_SIZE, }
7.  { CONTEXTS, }
10. { CONTIGUOUS, }
9.  { CREATE_IF }
12. { DEFAULT_NAME, }
14. { DEFERRED_WRITE, }
15. { DELETE_ON_CLOSE, }
15. { DIRECTORY_ENTRY, }
15. { ERASE_ON_DELETE, }
9.  { EXTENSION, }
19. { GLOBAL_BUFFER_COUNT, }
13. { MT_BLOCK_SIZE, }
19. { MT_CURRENT_POSITION, }
10. { MT_NOT_EOF }
13. { MT_PROTECTION, }
14. { MT_OPEN_REWIND, }
15. { MT_CLOSE_REWIND }
17. { MAX_RECORD_NUMBER, }
16. { MAXIMIZE_VERSION, }

```

```

4.      { NAME, }
8.      { NOBACKUP, }
16.     { NON_FILE_STRUCTURED }
17.     { OUTPUT_FILE_PARSE }
12.     { ORGANIZATION, }
5.      { OWNER, }
14.     { PRINT_ON_CLOSE, }
10.     { PROTECTION, }
10.     { READ_CHECK, }
8.      { REVISION, }
15.     { SEQUENTIAL_ONLY }
15.     { SUBMIT_ON_CLOSE, }
9.      { SUPERSEDE, }
9.      { TEMPORARY }
17.     { TRUNCATE_ON_CLOSE, }
14.     { USER_FILE_OPEN }
11.     { WINDOW_SIZE }
11.     { WRITE_CHECK, }

```

## ( JOURNALING PRIMARY )

```

11.     { AFTER_IMAGE, }
10.     { AFTER_NAME }
11.     { AUDIT_TRAIL, }
10.     { AUDIT_NAME }
12.     { BEFORE_IMAGE, }
11.     { BEFORE_NAME }
13.     { RECOVERY_UNIT, }

```

## ( KEY PRIMARY )

```

7.      { CHANGES, }
9.      { DATA_AREA, }
9.      { DATA_FILL, }
20.     { DATA_KEY_COMPRESSION, }
23.     { DATA_RECORD_COMPRESSION, }
10.     { DUPLICATES, }
10.     { INDEX_AREA, }
17.     { INDEX_COMPRESSION, }
10.     { INDEX_FILL, }
17.     { LEVEL_INDEX_AREA, }
4.      { NAMES, }
8.      { NULL_KEY, }
10.     { NULL_VALUE, }
6.      { PROLOG(UE) - 1ST 6 CHARS ONLY }
0.      { SEG_LENGTH, }
0.      { SEG_POSITION, }
0.      { SEG_TYPE, }

```

## ( RECORD PRIMARY )

```

10.     { BLOCK_SPAN, }
16.     { CARRIAGE_CONTROL, }
18.     { CONTROL_FIELD_SIZE, }
6.      { FORMAT, }
4.      { SIZE, }

```

## { SHARING PRIMARY }

```

6.      { DELETE }
3.      { GET }
11.     { MULTISTREAM }
8.      { PROHIBIT }
3.      { PUT }
6.      { UPDATE }
14.     { USER_INTERLOCK }

```

## { SYSTEM PRIMARY }

```

6.      { DEVICE, }
6.      { SOURCE, }
6.      { TARGET, }

);

```

{ +  
These are the maximum values of number-valued secondaries.  
- }

SECONDARY\_MAX := (

```

{ RESERVE 0 }      0.      { DUMMY_SECONDARY$, }

```

## { ACCESS PRIMARY }

```

0.      { BLOCK_IO$ }
0.      { DELETES }
0.      { GETS }
0.      { PUTS }
0.      { RECORD_IO$ }
0.      { TRUNCATES }
0.      { UPDATES }

```

## { ACL PRIMARY }

```

0.      { ENTRY }

```

## { ANALYSIS\_OF\_AREA PRIMARY }

```

0.      { RECLAIMED_SPACE }

```

## { ANALYSIS\_OF\_KEY PRIMARY }

```

0.      { DATA_FILLS, }
0.      { DATA_KEY_COMPRESSION, }
0.      { DATA_RECORD_COMPRESSION, }
0.      { DATA_RECORD_COUNT, }
0.      { DATA_SPACE_OCCUPIED, }
0.      { DELETIONS, }
0.      { DEPTH, }
0.      { DUPLICATES_PER_SIDR, }
0.      { INDEX_COMPRESSION, }
0.      { INDEX_FILLS, }

```

```

0.      { INDEX SPACE OCCUPIED }
0.      { LEVEL RECORD COUNT }
0.      { MEAN DATA LENGTH }
0.      { MEAN INDEX LENGTH }
0.      { RANDOM ACCESSES }
0.      { RANDOM INSERTS }
0.      { SEQUENTIAL ACCESSES }

```

## ( AREA PRIMARY )

```

EDFSC_1GIGA, { ALLOCATIONS }
0.          { BEST TRY CONTIGUOUS }
BKTSC_MAXBKTS12, { BUCKET_SIZES }
0.          { CONTIGUOUS }
0.          { EXACT POSITIONINGS }
EDFSC_1GIGA, { EXTENSIONS }
16777215,    { POSITIONS }
65535,      { VOLUMES }

```

## ( CONNECT PRIMARY )

```

0.      { ASYNCHRONOUS }
0.      { BLOCK IO }
EDFSC_1GIGA, { BUCKET CODE }
EDFSC_1GIGA, { CONTEXT }
0.      { END OF FILE }
0.      { FILE BUCKETS }
0.      { FAST DELETE }
255.    { KEY OF REFERENCE }
0.      { KEY GREATER EQUAL }
0.      { KEY GREATER THAN }
0.      { KEY LIMIT }
0.      { LOCATE MODE }
0.      { LOCK ON READ }
0.      { LOCK ON WRITE }
0.      { MANUAL UNLOCKING }
255.    { MULTIBLOCK COUNT }
255.    { MULTIBUFFER COUNT }
0.      { NOLOCK }
0.      { NONEXISTENT RECORD }
0.      { READ AHEAD }
0.      { READ REGARDLESS }
0.      { TIMEOUT ENABLE }
255.    { TIMEOUT PERIOD }
0.      { TRUNCATE ON PUT }
0.      { TT CANCEL CONTROL 0 }
0.      { TT-UPCASE-INPUT }
0.      { TT-PROMPT }
0.      { TT-PURGE TYPE AHEAD }
0.      { TT-READ-NOECHO }
0.      { TT-READ-NOFILTER }
0.      { UPDATE IF }
0.      { WAIT FOR RECORD }
0.      { WRITE BEHIND }

```

## ( DATE PRIMARY )



```
0.      { AFTER_IMAGE, }
0.      { AFTER_NAME }
0.      { AUDIT-TRAIL, }
0.      { AUDIT-NAME }
0.      { BEFORE_IMAGE, }
```

[illegible]



(FALSE, TRUE, FALSE, FALSE).  
 (FALSE, FALSE, FALSE, TRUE).  
 (FALSE, TRUE, FALSE, FALSE).  
 (FALSE, FALSE, FALSE, TRUE).  
 (FALSE, FALSE, FALSE, TRUE).  
 (FALSE, TRUE, FALSE, FALSE).  
 (FALSE, FALSE, FALSE, FALSE).  
 (FALSE, TRUE, FALSE, FALSE).

KEY: STR, NUM, QUAL, SW

```
{ ALLOCATIONS, }
{ BEST TRY CONTIGUOUS, }
{ BUCKET SIZES, }
{ CONTIGUOUS, }
{ EXACT POSITIONINGS, }
{ EXTENSIONS, }
{ POSITIONS, }
{ VOLUMES, }
```

[illegible]

- }

## { CONNECT PRIMARY }

(FALSE,FALSE,FALSE,TRUE).	{ ASYNCHRONOUS }
(FALSE,FALSE,FALSE,TRUE).	{ BLOCK IO }
(FALSE,TRUE,FALSE,FALSE).	{ BUCKET CODE }
(FALSE,TRUE,FALSE,FALSE).	{ CONTEXT }
(FALSE,FALSE,FALSE,TRUE).	{ END OF FILE }
(FALSE,FALSE,FALSE,TRUE).	{ FILE_BUCKETS }
(FALSE,FALSE,FALSE,TRUE).	{ FAST_DELETE }
(FALSE,TRUE,FALSE,FALSE).	{ KEY_OF_REFERENCE }
(FALSE,FALSE,FALSE,TRUE).	{ KEY_GREATER_EQUAL }
(FALSE,FALSE,FALSE,TRUE).	{ KEY_GREATER_THAN }
(FALSE,FALSE,FALSE,TRUE).	{ KEY_LIMIT }
(FALSE,FALSE,FALSE,TRUE).	{ LOCATE MODE }
(FALSE,FALSE,FALSE,TRUE).	{ LOCK_ON_READ }
(FALSE,FALSE,FALSE,TRUE).	{ LOCK_ON_WRITE }
(FALSE,FALSE,FALSE,TRUE).	{ MANUAL_UNLOCKING }
(FALSE,TRUE,FALSE,FALSE).	{ MULTIBLOCK_COUNT }
(FALSE,TRUE,FALSE,FALSE).	{ MULTIBUFFER_COUNT }
(FALSE,FALSE,FALSE,TRUE).	{ NOLOCK }
(FALSE,FALSE,FALSE,TRUE).	{ NONEXISTENT_RECORD }
(FALSE,FALSE,FALSE,TRUE).	{ READ_AHEAD }
(FALSE,FALSE,FALSE,TRUE).	{ READ_REGARDLESS }
(FALSE,FALSE,FALSE,TRUE).	{ TIMEOUT_ENABLE }
(FALSE,TRUE,FALSE,FALSE).	{ TIMEOUT_PERIOD }
(FALSE,FALSE,FALSE,TRUE).	{ TRUNCATE_ON_PUT }
(FALSE,FALSE,FALSE,TRUE).	{ TT_CANCEL_CONTROL_0 }
(FALSE,FALSE,FALSE,TRUE).	{ TT_UPCASE_INPUT }
(FALSE,FALSE,FALSE,TRUE).	{ TT_PROMPT }
(FALSE,FALSE,FALSE,TRUE).	{ TT_PURGE_TYPE_AHEAD }
(FALSE,FALSE,FALSE,TRUE).	{ TT_READ_NOECHO }
(FALSE,FALSE,FALSE,TRUE).	{ TT_READ_NOFILTER }
(FALSE,FALSE,FALSE,TRUE).	{ UPDATE_IF }
(FALSE,FALSE,FALSE,TRUE).	{ WAIT_FOR_RECORD }
(FALSE,FALSE,FALSE,TRUE).	{ WRITE_BEHIND }

## { DATE PRIMARY }

(TRUE,FALSE,FALSE,FALSE).	{ BACKUPS, }
(TRUE,FALSE,FALSE,FALSE).	{ CREATIONS, }
(TRUE,FALSE,FALSE,FALSE).	{ EXPIRATIONS, }
(TRUE,FALSE,FALSE,FALSE).	{ REVISIONS, }

## { FILE PRIMARY }

(FALSE,TRUE,FALSE,FALSE).	{ ALLOCATION, }
(FALSE,FALSE,FALSE,TRUE).	{ BEST_TRY_CONTIGUOUS, }
(FALSE,TRUE,FALSE,FALSE).	{ BUCKET_SIZE, }
(FALSE,TRUE,FALSE,FALSE).	{ CLUSTER_SIZE, }
(FALSE,TRUE,FALSE,FALSE).	{ CONTEXTS }
(FALSE,FALSE,FALSE,TRUE).	{ CONTIGUOUS, }
(FALSE,FALSE,FALSE,TRUE).	{ CREATE_IF }
(TRUE,FALSE,FALSE,FALSE).	{ DEFAULT_NAME, }
(FALSE,FALSE,FALSE,TRUE).	{ DEFERRED_WRITE, }



```
{ DELETE ON CLOSE, }
{ DIRECTORY ENTRY, }
{ ERASE ON DELETE, }
{ EXTENSION, }
{ GLOBAL BUFFER COUNT, }
{ MT_BLOCK SIZE, }
{ MT_CURRENT POSITION, }
{ MT_NOT_EOF, }
{ MT_PROTECTION, }
{ MT_OPEN REWIND, }
{ MT_CLOSE REWIND, }
{ MAX RECORD NUMBER, }
{ MAXIMIZE_VERSION, }
{ NAME, }
{ NOBACKUP, }
{ NON FILE STRUCTURED, }
{ OUTPUT FILE PARSE, }
{ ORGANIZATION, }
{ OWNER, }
{ PRINT ON CLOSE, }
{ PROTECTION, }
{ READ CHECK, }
{ REVISION, }
{ SEQUENTIAL ONLY, }
{ SUBMIT ON CLOSE, }
{ SUPERSEDE, }
{ TEMPORARY, }
{ TRUNCATE ON CLOSE, }
{ USER FILE OPEN, }
{ WINDOW SIZE, }
{ WRITE CHECK, }
```

{ JOURNAL PRIMARY }

```
{ AFTER_IMAGE }  
{ AFTER_NAME }  
{ AUDIT_TRAIL }  
{ AUDIT_NAME }  
{ BEFORE_IMAGE }  
{ BEFORE_NAME }  
{ RECOVERY_UNIT }
```

(FALSE,FALSE,FALSE,TRUE).  
(FALSE,TRUE,FALSE,FALSE).  
(FALSE,TRUE,FALSE,FALSE).  
(FALSE,FALSE,FALSE,TRUE).  
(FALSE,FALSE,FALSE,TRUE).  
(FALSE,FALSE,FALSE,TRUE).  
(FALSE,TRUE,FALSE,FALSE).  
(FALSE,FALSE,FALSE,TRUE).

```
{ CHANGES, }
{ DATA_AREA, }
{ DATA_FILL, }
{ DATA_KEY_COMPRESSION, }
{ DATA_RECORD_COMPRESSION, }
{ DUPLICATES, }
{ INDEX_AREA, }
{ INDEX_COMPRESSION, }
```

[illegible]

```
{ INDEX_FILL, }
{ LEVEL_INDEX_AREA, }
{ NAMES, }
{ NULL_KEY, }
{ NULL_VALUE, }
{ PROLOGUE, }
{ SEG_LENGTH, }
{ SEG_POSITION, }
{ SEG_TYPE, }
```

```
(FALSE,FALSE,FALSE,TRUE).
(FALSE,FALSE,TRUE,FALSE).
(FALSE,TRUE,FALSE,FALSE).
(FALSE,FALSE,TRUE,FALSE).
(FALSE,TRUE,FALSE,FALSE).
```

```
{ BLOCK_SPAN, }
{ CARRIAGE_CONTROL, }
{ CONTROL_FIELD_SIZE, }
{ FORMAT, }
{ SIZE, }
```

KEY: STR, NUM, QUAL, SW

[illegible]

```
{ DELETE }
{ GET }
{ MULTISTREAM }
{ PROHIBIT }
{ PUT }
{ UPDATE }
{ USER_INTERLOCK }
```

```
(TRUE,FALSE,FALSE,FALSE),
(FALSE,FALSE,TRUE,FALSE),
(FALSE,FALSE,TRUE,FALSE)
```

```
{ DEVICE, }
{ SOURCE, }
{ TARGET, }
```

) ;

[illegible]

```
{ +
This is the QTAB array, which controls the asking and processing of questions.
- }
```

```
QTAB := (
```

```
{ +
QUESTION OFFSET
ANSWER_CLASS,      DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- }
{ EDFSK_DATA_FILE_NAME }
{ STRING_ANSWER,    TRUE,          0,           0,           0,           0,           0),
{ EDFSK_FDL_TITLE }
{ STRING_ANSWER,    TRUE,          0,           0,           0,           0,           0),
{ EDFSK_KEY_NAME }
{ STRING_ANSWER,    TRUE,          0,           0,           0,           0,           0),
{ EDFSK_ANALYSIS }
{ STRING_ANSWER,    TRUE,          0,           0,           0,           0,           0),
{ EDFSK_OUTPUT }
{ STRING_ANSWER,    TRUE,          0,           0,           0,           0,           0),
{ EDFSK_DATA_KEY_COMP }
{ REAL_ANSWER,      TRUE,          0,          -99,          99,          0,           0),
{ EDFSK_DATA_RECORD_COMP }
{ REAL_ANSWER,      TRUE,          0,          -99,          99,          0,           0),
{ EDFSK_INDEX_RECORD_COMP }
{ REAL_ANSWER,      TRUE,          0,          -99,          99,          0,           0),
```

```
{ +
QUESTION OFFSET
ANSWER_CLASS,      DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- }
{ EDFSK_KEY_COMP_WANTED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_YES,    0,           0,           0,           0),
{ EDFSK_REC_COMP_WANTED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_YES,    0,           0,           0,           0),
{ EDFSK_IDX_COMP_WANTED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_YES,    0,           0,           0,           0),
{ EDFSK_ASCENDING_ADDED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
{ EDFSK_ASCENDING_LOAD }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
{ EDFSK_BLOCK_SPAN }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_YES,    0,           0,           0,           0),
{ EDFSK_CONFIRM }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
{ EDFSK_SEGMENTED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
{ EDFSK_GLOBAL_WANTED }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
```

```
{ +
QUESTION OFFSET
ANSWER_CLASS,      DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- }
{ EDFSK_KEY_CHANGES }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_YES,    0,           0,           0,           0),
{ EDFSK_KEY_DIST }
{ BOOLEAN_ANSWER,   TRUE,          EDFSK_NO,     0,           0,           0,           0),
{ EDFSK_KEY_DUPS }
```

```

(BOOLEAN ANSWER,      TRUE,      EDF$K_NO,      0,      0,      0,      0),
( EDF$K_RETURN )
(BOOLEAN ANSWER,      TRUE,      0,      0,      0,      0,      0),
( EDF$K_CLUSTER_SIZE )
(INTEGER ANSWER,      TRUE,      3,      1,      EDF$C_1GIGA,      0,      0),
( EDF$K_ACTIVE_KEY )
(INTEGER ANSWER,      TRUE,      0,      0,      0,      0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,          DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- )
( EDF$K_ADDED_COUNT )
(INTEGER ANSWER,      TRUE,      0,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_ADDED_COUNT_LOW )
(INTEGER ANSWER,      TRUE,      0,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_ADDED_COUNT_HIGH )
(INTEGER ANSWER,      TRUE,      100000,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_BLOCKS_IN_BUCKET )
(INTEGER ANSWER,      TRUE,      32,      1,      BKT$C_MAXBKTSIZ,      0,      0),
( EDF$K_BUCKET_WEIGHT )
(KEYWORD ANSWER,      TRUE,      EDF$K_FLATTER_FILES,      0,      0,      0,      0),
( EDF$K_CARR_CTRL )
(KEYWORD ANSWER,      TRUE,      FDL$C_CR,      0,      0,      0,      0),
( EDF$K_CONTROL_SIZE )
(INTEGER ANSWER,      TRUE,      2,      1,      255,      0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,          DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- )
( EDF$K_CURRENT_FUNCTION )
(KEYWORD ANSWER,      TRUE,      EDF$K_HELP,      0,      0,      0,      0),
( EDF$K_DESIGN_CYCLE )
(KEYWORD ANSWER,      TRUE,      EDF$K_WP,      0,      0,      0,      0),
( EDF$K_DESIRED_FILL )
(INTEGER ANSWER,      TRUE,      100,      0,      100,      0,      0),
( EDF$K_FILL_LOW )
(INTEGER ANSWER,      TRUE,      50,      0,      100,      0,      0),
( EDF$K_FILL_HIGH )
(INTEGER ANSWER,      TRUE,      100,      0,      100,      0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,          DEFAULT_OK,      DEFAULT,      LOW_BOUND,      HIGH_BOUND,      KEY_TABLE,      STATE_TABLE
- )
( EDF$K_GLOBAL_COUNT )
(INTEGER ANSWER,      FALSE,      0,      0,      65535,      0,      0),
( EDF$K_GRANULARITY )
(KEYWORD ANSWER,      TRUE,      EDF$K_THREE,      0,      0,      0,      0),
( EDF$K_INITIAL_COUNT )
(INTEGER ANSWER,      FALSE,      0,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_INITIAL_COUNT_LOW )
(INTEGER ANSWER,      TRUE,      0,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_INITIAL_COUNT_HIGH )
(INTEGER ANSWER,      TRUE,      100000,      0,      EDF$C_1GIGA,      0,      0),
( EDF$K_KEY_POSITION )
(INTEGER ANSWER,      TRUE,      0,      0,      EDF$K_MAXRECSIZ,      0,      0),
( EDF$K_KEY_LOW )

```



```

(INTEGER ANSWER,      TRUE,      1,      0,      0,      0,      0),
( EDF$K KEY HIGH )
(INTEGER ANSWER,      TRUE,      255,    0,      0,      0,      0),
( EDF$K KEY SIZE )
(INTEGER ANSWER,      FALSE,     0,      0,      0,      0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,        DEFAULT_OK,  DEFAULT,  LOW_BOUND,  HIGH_BOUND,  KEY_TABLE,  STATE_TABLE
- )
( EDF$K KEY TYPE )
(KEYWORD ANSWER,      TRUE,      FDL$C_STG,      0,      0,      0,      0),
( EDF$K LOAD METHOD )
(KEYWORD ANSWER,      TRUE,      EDF$K_FAST_CONVERT, 0,      0,      0,      0),
( EDF$K MAX RECORD_SIZE )
(INTEGER ANSWER,      FALSE,     0,      0,      0,      0,      0),
( EDF$K MEAN RECORD_SIZE )
(INTEGER ANSWER,      FALSE,     0,      1,      EDF$K_MAXRECSIZ, 0,      0),
( EDF$K NUMBER DUPS )
(INTEGER ANSWER,      TRUE,      0,      0,      EDF$C_1GIGA, 0,      0),
( EDF$K NUMBER KEYS )
(INTEGER ANSWER,      TRUE,      1,      1,      255,    0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,        DEFAULT_OK,  DEFAULT,  LOW_BOUND,  HIGH_BOUND,  KEY_TABLE,  STATE_TABLE
- )
( EDF$K NUMBER RECORDS )
(INTEGER ANSWER,      FALSE,     0,      0,      EDF$C_1GIGA, 0,      0),
( EDF$K PROLOGUE_VERSION )
(INTEGER ANSWER,      TRUE,      3,      0,      3,      0,      0),
( EDF$K PROMPTING )
(KEYWORD ANSWER,      TRUE,      EDF$K_FULL, 0,      0,      0,      0),
( EDF$K RECORD FORMAT )
(KEYWORD ANSWER,      TRUE,      FDL$C_VAR, 0,      0,      0,      0),
( EDF$K RESPONSES )
(KEYWORD ANSWER,      TRUE,      EDF$K_AUTO, 0,      0,      0,      0),
( EDF$K SCRIPT OPTION )
(KEYWORD ANSWER,      FALSE,     0,      0,      0,      0,      0),
( EDF$K SET FUNCTION )
(KEYWORD ANSWER,      FALSE,     0,      0,      0,      0,      0),
( EDF$K SIZE LOW )
(INTEGER ANSWER,      TRUE,      1,      1,      EDF$K_MAXRECSIZ, 0,      0),
( EDF$K SIZE HIGH )
(INTEGER ANSWER,      TRUE,      1000, 1,      EDF$K_MAXRECSIZ, 0,      0),
( EDF$K SURFACE_OPTION )
(KEYWORD ANSWER,      TRUE,      EDF$K_LINE_SURFACE, 0,      0,      0,      0),
( +
QUESTION OFFSET
ANSWER_CLASS,        DEFAULT_OK,  DEFAULT,  LOW_BOUND,  HIGH_BOUND,  KEY_TABLE,  STATE_TABLE
- )
( EDF$K TEST PRIMARY )
(KEYWORD ANSWER,      TRUE,      FDL$C_FILE, 0,      0,      0,      0),
( EDF$K TEST SECONDARY )
(OBJECT ANSWER,      FALSE,     0,      0,      0,      0,      0),
( EDF$K TEST SECONDARY_VALUE )
(OBJECT ANSWER,      FALSE,     0,      0,      0,      0,      0)

```

{ PROT\_MASK }

[illegible]





0124 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY